UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Aljoša Balažic

Spletna aplikacija za označevanje polipov na podvodnih fotografijah

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: doc. dr. Matjaž Kukar

Ljubljana, 2014

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Razvijte spletno aplikacijo, ki bo omogočala upravljanje in označevanje polipov na zajetih podvodnih fotografijah. Zaledni del aplikacije naj omogoča učinkovito shranjevanje in upravljanje fotografij ter oznak, interaktivni del pa naj bo sodobna interaktivna spletna aplikacija. Omogočite verzioniranje naborov oznak in predpripravo za priklop avtomatskega označevalnika slik. Osredotočite se na preprostost uporabniške izkušnje, ki bo omogočala delo tudi tehnično manj usposobljenim kadrom, ter jo ustrezno ovrednotite.

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Spodaj podpisani Aljoša Balažic, z vpisno številko **63040449**, sem avtor diplomskega dela z naslovom:

Spletna aplikacija za označevanje polipov na podvodnih fotografijah

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Matjaža Kukarja,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 25. septembra 2014 Podpis avtorja:

Zahvaljujem se vsem, ki so verjeli vame in me vzpodbujali, da je nastala ta diplomska naloga. Hvala staršem za podporo v celotnem času študija, prijateljem in sodelavcem, ki so prenašali moje godrnjanje in predvsem mentorju dr. Matjažu Kukarju za vso strokovno pomoč in veliko mero potrpežljivosti.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
HTML	Hyper Text Markup Language	označevalni jezik za izdelavo spletnih strani
CSS	Cascading Style Sheets	stilski jezik za urejanje videza html dokumenta
AJAX	Asynchronous JavaScript and XML	skupina tehnologij za ustvarjanje dinamičnih spletnih strani
GPS	Global Positioning System	sistem globalnega pozicioniranja
URL	Uniform Resource Locator	naslov spletnih strani v svetovnem spletu
EXIF	Exchangeable image file format	skupek prenosljivih podatkov o fotografiji
MYSQL	My Structured Query Language	odprtokodni sistem za upravljanje zbirk podatkov
JQUERY		knjižnica za skriptni jezik JavaScript, ki prinaša zbirko funkcij za hitrejši razvoj spletnih aplikacij

Povzetek

Za morske biologe je pomembno napovedovanje pričakovanega števila meduz, česar se lotevajo s fotografiranjem polipov na morskem dnu, iz katerih se pozneje razvijejo meduze. Ročno štetje polipov na fotografijah je zamudno in naporno delo, zato se bo v drugem delu aplikacije implementiral sistem za avtomatsko označevanje in štetje. Ker pa vemo da avtomatika ni popolna, je pogosto potrebna človeška intervencija.

V diplomski nalogi je tako predstavljena sodobna spletna aplikacija, ki vsebuje komponente in obrazce za masovno nalaganje slik, prikaz in osnovno obdelavo slik in oznak na slikah, korekcijo (potrjevanje, zavračanje, dodajanje in brisanje) nad že avtomatsko zaznanimi oznakami, ter ustrezno interakcijo med odjemalcem, strežnikom in podatkovno bazo. S strani uporabnika popravljene oznake so tudi podlaga za nadaljnje pristope za strojno učenje avtomatskega zaznavanja. V prvem delu diplomske naloge so opisane uporabljene tehnologije in orodja. Sledi zajem zahtev, načrtovanje in implementacija, v zadnjem delu pa je predstavljena sama uporaba in učinkovitost aplikacije, spoznanja do katerih smo prišli tekom izdelave, ter možnosti za nadaljnji razvoj.

Ključne besede: spletna aplikacija, polip, fotografija, meduza, označevanje

Abstract

Predicting the expected number of jellyfish is very important for marine biologists and the approach is taken by photographing polyps at the bottom of the sea which later reproduce to jellyfish. Counting polyps by hand is a long and tedious task, therefore the second part of this web application will be implementation of automatic counting and marking system. As we know such systems are not perfect and human intervention is often required.

This thesis introduces a modern web application that includes components and interfaces for solving multiple image uploads, previewing and processing of images, including reviewing (adding, deleting, confirming, denying) automatically detected polyp markings and appropriate interaction between client, server and database. The user reviewed markings are the basis for further machine learning of automatic detection. The first part of the thesis describes technologies used, followed by application requirements. The last part introduces the use, efficiency of the application and the results that we acquired during the process of development.

Keywords: web application, polyp, photograph, mark, medusa, marking

Poglavje 1 Uvod

Da bi bolje razumeli populacijo meduz in njihovo periodično pojavljanje, morski biologi preučujejo njihovo razmnoževanje v različnih obdobjih. Takšne raziskave lahko pripomorejo pri razlagi, kako različne okoliščine, kot so temperatura vode in slanost, vplivajo na število meduz. Meduze se razvijejo iz polipov, ki so, za razliko od plavajočih meduz, pritrjeni na morsko dno. V obdobjih cvetenja polipov tako potapljači Morske biološke postaje Piran na različnih mestih fotografirajo podmorske površine, da bi lahko prešteli polipe in posledično predvideli število meduz.

Število polipov na slikah je lahko veliko, zato je ročno štetje časovno potratno in predvsem zelo naporno. Avtomatizacija tega procesa bi zato občutno pohitrila takšno študijo. Vemo pa, da je avtomatsko zaznavanje pogosto nenatančno oziroma nepopolno, zato so potrebne korekcije s strani človeka. Cilj diplomske naloge je tako izdelati spletno aplikacijo, preko katere bo s fotoaparata možno naložiti slike, sprožiti avtomatsko detekcijo polipov (sama izvedba avtomatike ne sodi v to diplomsko nalogo), oznake pa kasneje popravljati in urejati. Aplikacija bo zato morala vsebovati enostaven urejevalnik, podoben namizni aplikaciji za urejanje slik. Vsi podatki se bodo shranjevali v podatkovno bazo, ki bo med drugim služila kot vir za strojno učenje postopkov avtomatičnega označevanja.

Pri razvoju aplikacije se bomo trudili, da bo za uporabnika karseda prijazna in enostavna, ter z vidika dostopnosti tudi varna. Ker se je uporaba spleta v zadnjih letih zelo povečala, bo temeljila na številnih spletnih tehnologijah in bo tako bolj dostopna.

Na začetku diplomskega dela je na kratko predstavljena Morska biološka postaja Piran in njihove želje oziroma potrebe glede aplikacije. Sledi predstavitev izbranih tehnologij, ki smo jih pri izdelavi uporabljali. V osrednjem delu je razloženo delovanje in razvoj aplikacije. Na koncu predstavimo še testiranje in na kakšen način je potrebno povezati aplikacijo z avtomatskim delom, ki bo razvit kasneje.

Poglavje 2 O aplikaciji

2.1 Predstavitev Morske biološke postaje

Morska biološka postaja Piran (MBP) [14] je bila ustanovljena leta 1969 kot enota Nacionalnega inštituta za biologijo iz Ljubljane. Podobno kot njen predhodnik Zavod za raziskovanje morja, ki je deloval že nekaj let poprej, je MBP nastala zaradi naraščajočega zanimanja za morje, a tudi zaradi dejanskih potreb - predvsem skrbi za okolje. Sprva je bila postaja usmerjena v raziskovanje lokalne flore in favne, sčasoma pa je prerasla v večjo enoto, ki je prenesla težišče dela tudi na ekološke, polucijske in danes že pretežno interdisciplinarne raziskave morja.

Morska biološka postaja danes deluje kot edina raziskovalna skupina v Sloveniji, ki se ukvarja z znanstvenim in strokovnim delom, v celoti vezanim na morje. V povezavi z vrhunskimi tujimi centri predstavlja žarišče razvoja na tem področju. Skozi tri desetletja svojega razvoja je MBP prerasla ozek okvir terenske morske izpostave Nacionalnega inštituta za biologijo, se programsko, kadrovsko in materialno močno okrepila in danes, tesno vpeta v domači in mednarodni raziskovalni prostor, predstavlja pomemben vezni člen narodove naravoslovne biti z morjem.

Dejavnosti:

- temeljne raziskave – cilj je poglobiti znanje o dinamiki obalnih morskih ekosistemov,
- aplikativne raziskave - znanstvene in strokovne ekspertize za številne naročnike, predvsem s področja ekologije, varstva okolja, prostorskega planiranja itd.,
- stalno spremljanje kakovosti morja - z namenom preučiti vplive človekove dejavnosti na ekološke razmere, ter zbiranja informacij za potrebe prostorskega načrtovanja, turizma in naravovarstvenikov,
- pedagoška dejavnost - obsega predavanja in seminarje s tematiko spoznavanja naravnih značilnosti morja, njegove ekologije in varstva,

- program svetovanja - svetovanja glede upravljanja z obalnim pasom in morjem za obalne občine in podjetja, svetovanja v izobraževalnih programih ipd.

2.2 Uporabniške zahteve

V okviru diplomske naloge je potrebno razviti spletno aplikacijo, ki bo morskim biologom omogočala poganjanje, pregled in popravljanje avtomatsko zaznanih polipov na podvodnih fotografijah. Zahteve so bile pridobljene v sodelovanju s končnim uporabnikom - Morsko biološko postajo Piran. Spodaj pa so razčlenjene vse zahteve po sklopih.

Sistem avtentikacije:

- pregled nad uporabniki,
- dodajanje in brisanje uporabnikov,
- urejanje uporabnika (e-pošta, ime, priimek, geslo …),
- obrazec za prijavo v sistem,
- možnost odjave iz sistema,
- obrazec za pozabljeno geslo.

Stran s seznamom zajetih podvodnih fotografij:

- možnost masovnega nalaganja novih fotografij po sistemu vleci in spusti ali prek okna za izbiro datotek iz določene mape,
- izpis fotografij v seznamu,
- številčenje strani v primeru, da je v seznamu veliko fotografij,
- filtriranje fotografij po datumu nastanka fotografije, z možnostjo izbire začetnega in končnega datuma iz koledarčka,
- predogled pomanjšane sličice fotografije in možnost odpiranja izvorne v novem zavihku brskalnika,

- spreminjanje naslova fotografije,
- prikaz EXIF [4] podatkov s fotografije (EXIF so podatki, ki jih k datoteki slike zapiše naprava, ki sliko zajame in tipično vključujejo datum nastanka, model fotoaparata, tip slike, geografske koordinate, ločljivost …),
- povezavo do urejanja, ter prikaz velike in majhne sličice izbranega območja na sliki, znotraj katerega se polipi iščejo,
- izpis informacij oznak polipov, ki vsebujejo zaporedno številko verzije, tip (uporabniški ali avtomatski), število oznak polipov in povezavo do urejanja oz. pregleda verzije
- možnost ustvarjanja in brisanja verzij oznak polipov – tako avtomatskih, kot uporabniških verzij, ki se ustvarijo za avtomatskimi in služijo popravljanju avtomatskih,
- brisanje fotografije in vseh pripadajočih verzij in oznak.

Stran za izbiro območja na fotografiji, v katerem se iščejo polipi:

- privzeto se po nalaganju slike območje nastavi čez celotno sliko,
- prikaz informacij o izbranem območju, ki se naj posodablja sprotno,
- shranjevanje izbranega območja in generiranje slike območja v naravni velikosti za uporabo v urejevalniku oznak,
- izbiro območja na pomanjšani sliki z vlečenjem miškinega kazalca in zatemnjevanjem delov, ki niso zajeti na sliki.

Stran za pregled in urejanje označenih polipov na fotografiji:

- orodna vrstica, z vsemi potrebnimi orodji za delo z oznakami na fotografiji,
- okno s prikazom fotografije v ozadju in oznak v ospredju,

- orodje za povečavo fotografije in oznak, s pripadajočim procentualnim izpisom v orodni vrstici, drsnikom in ikonama povečaj, pomanjšaj za spreminjanje stopnje povečave, ter možnost spreminjanja z miškinim koleščkom,
- vodoravni in navpični drsnik, ki se pokažeta v primeru da je slika večja od velikosti platna v katerem prikazujemo fotografijo, z možnostjo premikanja ob uporabi miškinih koleščkov,
- orodje za izbiro oznak, premikanje, spreminjanje velikosti, ter brisanje,
- orodje za izbiro pri novi ali spreminjanje obstoječega tipa oznake (krog, elipsa, kvadrat, pravokotnik, poligon),
- orodje za risanje novih oznak na podvodni fotografiji,
- orodje za spreminjanje statusa (potrjen, zavrnjen, nedefiniran) izbrane oznake,
- izpis verzij, ter možnost preklopa na drugo ali shranjevanja trenutne verzije.

Poglavje 3 Tehnologije in programska oprema

Pri izdelavi aplikacije smo uporabili številne tehnologije, ki omogočajo razvoj modernih spletnih strani. Na strani odjemalca smo v različnih brskalnikih odpirali HTML dokumente, ki so ob podpori JavaScript knjižnic jQuery in Paper.js, omogočili urejevalnik. Za očem prijeten izgled je poskrbel CSS. Na strani strežnika smo poganjali IIS (Internet Information Sevices) in programski jezik PHP, za hrambo podatkov pa je skrbela podatkovna baza MySQL. Ker vsak razvijalec potrebuje tudi dobro orodje, smo si za urejanje namestili Eclipse IDE.

3.1 HTML

HTML (angleško HyperText Markup Language, slovensko jezik za označevanje nadbesedila) [7] je označevalni jezik, ki je namenjen izdelavi spletnih strani. Predstavlja osnovo spletnega dokumenta in vsebuje zaporedje ukazov, ki definirajo strukturo in smiselne povezave vsebine (slika 3.1), ter povedo kako naj se vsebina prikaže. Ukazom drugače rečemo tudi značke in se vedno nahajajo med znakoma < in >. Poznamo dva tipa značk - samostoječe, ki ne potrebujejo zaključka (npr.
) in kombinacijo začetne in končne značke, med katerima se nahaja vsebina ali druge gnezdene značke. Značke imajo tudi atribute, npr. title (naslov), href (URL)… Tipične značke predstavljajo naslove, odstavke, tabele, sezname, vnosna polja ipd. Vsak HTML dokument je praviloma vedno sestavljen iz dveh glavnih delov. Glava (angl. head) dokumenta je prvi del, ki vsebuje meta podatke, povezave do skript in stilov, naslov dokumenta. Drugi del strani je telo (angl. body), ki zajema vidni del strani in je tipično razdeljen z elementi div, article, nav …

```
<!DOCTYPE html>
<html>
    <head>
        <title>Naslov spletnega mesta</title>
    </head>
    <body>
        <h1>Moj prvi naslov</h1>
        <p>Moj prvi odstavek.</p>
    </body>
</html>
```

Slika 3.1: Primer HTML dokumenta enostavne spletne strani

HTML5 je v času pisanja zadnja različica, ki prinaša nekatere pomembne novosti na področju večpredstavnosti. Med te spadajo elementi video, avdio in canvas. Platno (angl. canvas) ima pri tej diplomski nalogi eno od ključnih vlog in je podrobneje predstavljen v svojem podpoglavju.

3.2 CSS

CSS (angleško Cascade Style Sheet, slovensko kaskadne stilske podloge) [2] so podloge, predstavljene v obliki preprostega slogovnega jezika (slika 3.2), ki skrbi za prezentacijo spletnih strani. Omogoča, da definiramo obliko oziroma izgled posameznih HTML elementov in skupin ločeno od same HTML kode, kar poveča preglednost. Gre za ločitev strukture in vsebine spletne strani od izgleda. Tipične lastnosti, ki jih CSS definira so barve, ozadja, odmiki, razmiki, pozicije, pisave, velikosti … Določamo pa lahko tudi spremenjeni izgled elementa ob neki uporabniški aktivnosti oziroma dogodku, npr. premiku z miškinim kazalcem ali pritisku tipke. CSS pravila določamo na tri načine: tip elementa, id atribut, class atribut.

```
a {
    color: red;
}

a:hover {
    text-decoration: underline;
}

#logo {
    border: 1px solid black;
}

.center {
    text-align: center;
}
```

Slika 3.2: Primer CSS oblikovnih definicij

Velik problem CSS-ja v preteklosti je bila različna podpora in interpretacija brskalnikov, predvsem IE je razvijalcem povzročal velike preglavice, zaradi česar danes obstaja W3C [20] konzorcij (World Wide Web Consortium), ki skrbi za razvoj spletnih standardov. Zadnja verzija CSS3 je prinesla veliko novosti kot so animacije, tranzicije, možnost uporabe kateregakoli fonta, gradiente, zaobljene robove. Posebej pa velja omeniti medijske poizvedbe (angl. media-query), ki so omogočile nastanek odzivnih spletnih strani, ki se glede na velikost zaslona prikažejo drugače (npr. prilagojeno za mobilne telefone).

3.3 jQuery

jQuery [12] je knjižnica za skriptni jezik JavaScript, ki vsebuje vnaprej napisane funkcije, ki nam olajšajo delo tako, da je potrebno napisati manj kode, da bi dosegli isti oziroma večji učinek (slika 3.3). Rešuje tudi problem različne implementacije v brskalnikih in razvijalcu prihrani čas prilagajanja in testiranja kode. jQuery je uporabljen na 60% najbolj obiskanih spletnih strani in je najbolj popularna JavaScript knjižnica. Poenostavlja oz. omogoča:

- izbiranje in manipulacijo HTML elementov na osnovi CSS selektorjev
- upravljanje z dogodki na spletni strani,
- efekte in animacije,

- interakcijo AJAX,
- spreminjanje (filtriranje, brisanje, dodajanje …) elementov v DOM (Document Object Model).

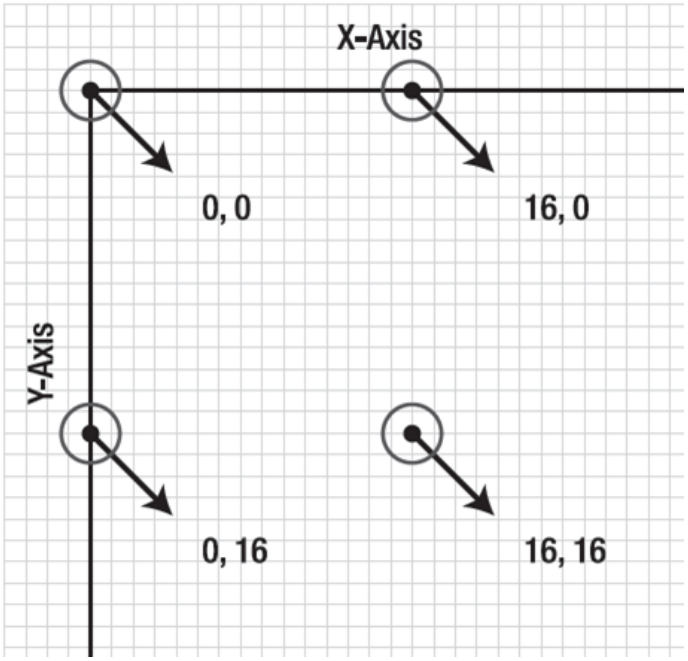
Osnovni ukaz, ki nakazuje, da se sklicujemo oziroma uporabljamo knjižnico jQuery je $, kateremu običajno sledi selektor, zapisan v oklepajih in dejanje, ki naj se izvrši nad izbrano množico elementov.

```
$(function() {
    // sproži ob kliku na povezavo
    $('a').click(function() {

        // prikaži vse naslove h1 ki imajo atribut class = hidden
        $('h1.hidden').fadeIn();

        // odstrani vse odstavke
        $('p').remove();
    });
})
```

Slika 3.3: Primer enostavnega jQuery programa

3.4 Canvas

Canvas (angl. platno) [1] je noviteta v HTML5, ki si jo lahko predstavljamo kot dvodimenzionalno mrežo (slika 3.4) oz. platno, ki omogoča dinamično risanje in prikazovanje grafike na spletni strani. Določiti mu je potrebno širino in višino, ter ID objekta, preko katerega delamo z njim. Za uporabo in nadzor nad izrisovanjem se uporablja JavaScript.



Slika 3.4: Platno kot koordinatna mreža

HTML5 canvas podpirajo modernejši brskalniki kot so Firefox 3, Chrome 2, Opera 10 in Internet Explorer 9, ter novejše različice. Za prikazovanje platno ne potrebuje dodatnih vtičnikov, kot je npr. Adobe Flash. Uporabimo ga lahko za:

- risanje diagramov,
- različne animacije in igre,
- grafe in razpredelnice,
- risalne aplikacije.

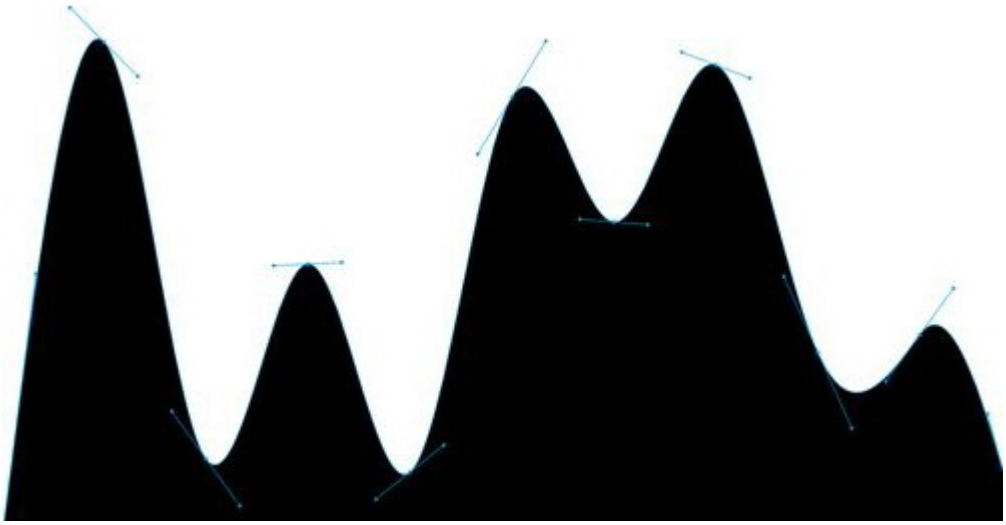
Platno je v tej spletni aplikaciji podlaga za prikaz slik in obdelavo oznak na slikah. Konkretneje je njegova naloga izris kvadratov, krogov, poligonov, ki prekrivajo polipe na slikah. Znati mora prikazati tudi premikanje, dodajanje, brisanje, barvanje in spreminjanje velikosti teh elementov.

3.5 Paper.js

Paper.js [16] je odprtokodna knjižnica za delo z vektorsko grafiko, ki za izris uporablja HTML5 canvas (platno). Temelji na skriptnem jeziku Scriptographer (Adobe Illustrator).

Glavna ideja je poenostavitev dela s platnom, tako da se razvijalcu ni potrebno toliko ukvarjat s tem kako bo neko stvar izrisal, ampak se lahko bolj posveti samim grafičnim elementom. Omogoča pa še:

- delo s plastmi (angl. layer), skupinami, potmi (angl. path), rastri, simboli itd.,
- avtomatski izris grafike, saj za ukaze za delo s platnom skrbi Paper.js,
- skriptni jezik PaperScript, ki je podoben JavaScriptu,
- vse oblike kot so kvadrat, krog, točka, elipsa ipd.,
- enostavne matematične operacije z vektorskimi in geometrijskimi objekti,
- upravljanje dogodkov ob uporabi miške in tipkovnice,
- ustvarjanje poti in spreminjanje njihovih krivulj in ročk …



Slika 3.5: Prikaz nekakšnih animiranih valov s pripadajočimi ročkami na uradni spletni strani knjižnice Paper.js

3.6 PHP

PHP (angleško PHP HyperText Preprocessor, slovensko predprocesor hiperteksta) [17] je razširjen odprtokodni programski jezik, ki se uporablja za razvoj dinamičnih strani, pa tudi kot generičen. PHP teče na spletnem strežniku (Apache, IIS in drugi), kjer jemlje PHP izvorno kodo za vhod in generira spletno stran kot izhod. Možno ga je uporabljati tudi v

ukaznem načinu. Programu, ki je napisan v PHP kodi, pravimo PHP skripta. PHP koda (slika 3.6) je praviloma zavita s posebnima oznakama <?php in ?>. Lahko jo vstavimo med HTML kodo, ali pa uporabimo kombinacijo z mehanizmom za upravljanje s predlogami (angl. template engine). Ko uporabnikov brskalnik prikaže spletno stran, ne ve da je napisana v PHP jeziku, saj je od strežnika, ki je predhodno procesiral PHP, dobil nazaj HTML. PHP tudi podpira dostop do večine podatkovnih baz kot so MySQL, Oracle, DB2, PostgreSQL in ostale.

```
<?php
    echo "<h1>Pisanje diplomske naloge je zabavno :)<h1>";

    for ($i = 1; $i < 5; $i++) {
        echo "<h2>To je podpoglavje $i.</h2>";
    }
?>
```

Slika 3.6: Primer preproste PHP skripte ki izpiše glavni naslov in štiri podnaslove

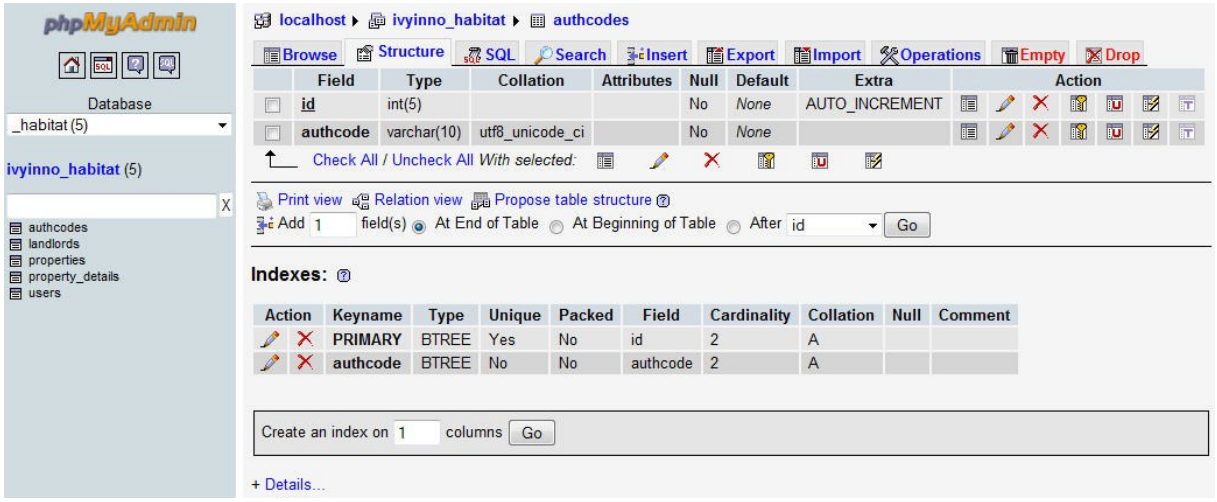
3.7 IIS / Apache

Ker je aplikacija razvita v programskem jeziku PHP, ki teče na spletnem strežniku, je bilo tak strežnik potrebno vzpostaviti. K sreči so v Windows 8 že priložili strežniški sistem IIS (Internet Information Services) [10], ki PHP podpira in ga je potrebno samo vklopit. Strežnik poskrbi za prejem zahtevkov, brskalniku pa vrača ustrezno HTML kodo, slike in ostale vire, ki so potrebni za delovanje strani. Druga možnost bi bil LAMP (Linux, Apache, MySQL, PHP) [13] paket za Windows, npr. XAMPP ali WAMP. Gre za prednastavljen paket, ki že vsebuje Apache, MySQL, PHP, phpMyAdmin in druga orodja, ter ga je mogoče zelo hitro namestiti.

Za lokalni razvoj je ta kombinacija sicer popolnoma zadostovala, za uporabo v produkciji pa je smiselno uporabiti operacijski sistem Linux, ki je naravno okolje za strežnik Apache in programski jezik PHP. Takšno okolje je tudi bolje zaščiteno (npr. ni virusov), stabilno in zmogljivo.

3.8 MySQL

MySQL [15] je sistem za upravljanje s podatkovnimi bazami. Gre za odprtokodno implementacijo relacijske podatkovne baze, kar pomeni, da so podatki shranjeni v ločenih tabelah in ne v enem velikem hranilniku, kar poveča hitrost. Za delo z MySQL se uporablja jezik SQL (Structured Query Language), ki je najbolj uporabljen standardiziran jezik za dostop do zbirk podatkov. Delovanje sistema MySQL je po principu odjemalec – strežnik. Sam po sebi MySQL nima GUI vmesnika za administracijo ali upravljanje s podatki, obstaja pa veliko število odjemalcev in vmesnikov za dostop. Najbolj priljubljen je verjetno phpMyAdmin, ki je dejansko kar spletna aplikacija (slika 3.7), napisana v programskem jeziku PHP.



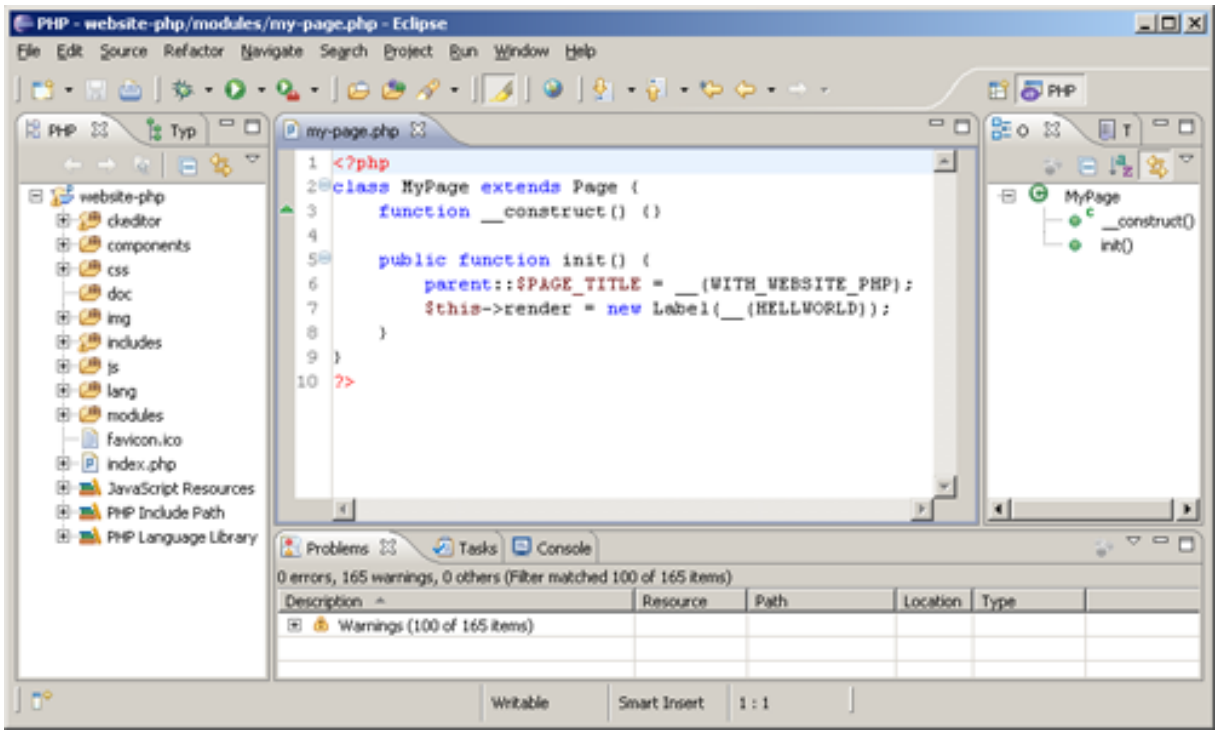
Slika 3.7: Urejanje podatkov in podatkovnih zbirk z grafičnim vmesnikom phpMyAdmin

V naši aplikaciji MySQL hrani podatke o oznakah polipov (tip, status, verzija, pozicija …), naloženih slikah in skrbi, da se v aplikacijo lahko prijavijo samo avtorizirani uporabniki (uporabniška imena, gesla).

3.9 Eclipse IDE

Za izdelavo vsakega programa potrebujemo urejevalnik v katerem spišemo kodo. Izbira je po navadi preferenca vsakega posameznika. Za diplomsko nalogo smo uporabili Eclipse IDE (slika 3.8) [3], ki je večjezično razvojno okolje, ki vključuje integrirano razvojno okolje (IDE) in možnost razširitev (plug-in) za programske jezike s katerimi delamo.

Za potrebe naše aplikacije Eclipse podpira vse uporabljene programske jezike – PHP, HTML, JavaScript, CSS. Potrebno pa je bilo naložiti Eclipse PDT razširitev (PHP Development Tools).



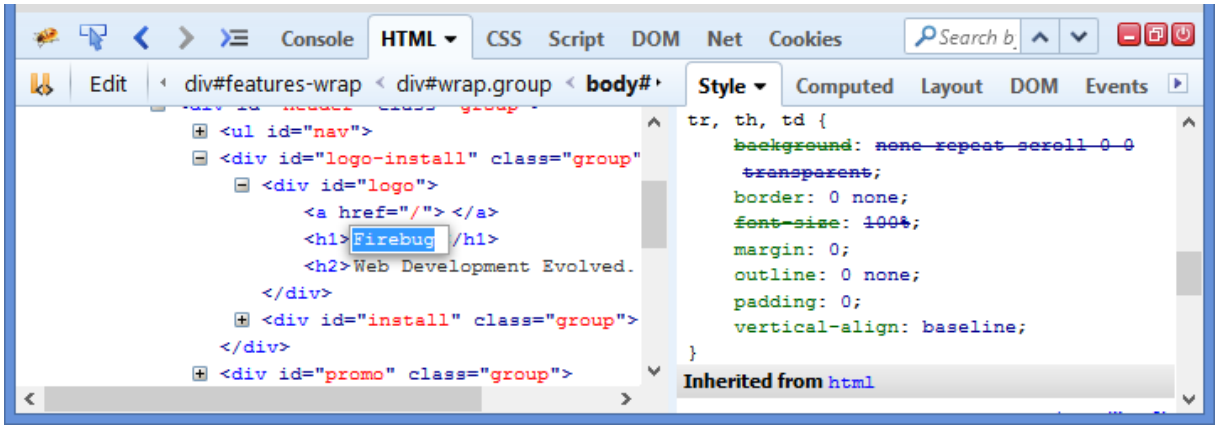
Slika 3.8: Integrirano razvojno okolje Eclipse IDE

3.10 Firebug

Firebug [6] je dodatek za spletni brskalnik Firefox, ki razvijalcu omogoča razhroščevanje, urejanje in nadziranje CSS, HTML, DOM, JavaScripta.

- JavaScript zavihek lahko npr. beleži napake, profilira klice funkcij in omogoča poganjanje poljubne JavaScript kode, ter omogoča vse funkcije tipičnega razhroščevalnika,
- v zavihku mreža lahko spremljamo karakteristike zahtevkov brskalnika (čas nalaganja, število, tip …) in tako analiziramo učinkovitost,
- CSS zavihek prikaže, na katere HTML elemente se aplicirajo CSS stili in omogoča spreminjanje stilov »v živo« (slika 3.9),

- v HTML zavihku se lahko sprehajamo po HTML strukturi, spremljamo CSS stile trenutno izbranega elementa, postavitev, dodajamo in odstranjujemo atribute …



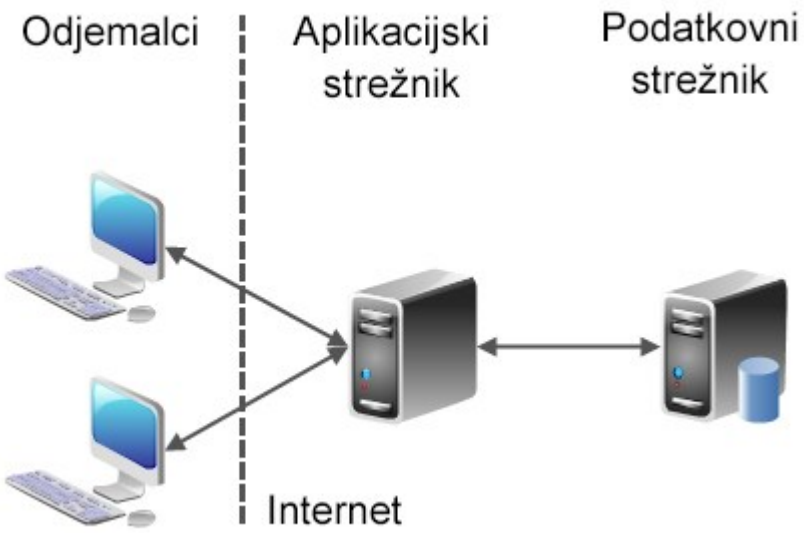
Slika 3.9: Primer razprte HTML strukture in prikaz apliciranih CSS stilov

Poglavje 4 Razvoj aplikacije

4.1 Arhitektura sistema

Pred začetkom razvoja aplikacije je bilo potrebno določiti tudi arhitekturo sistema. Za najprimernejšo se je izkazala tri-nivojska arhitektura (slika 4.1) [19], kar pomeni, da imamo:

- aplikacijski strežnik (strežnik, na katerem je nameščena uporabniška aplikacija), ki izvaja aplikacijsko logiko – v tem primeru skrbi za obdelavo podatkov, procesiranje PHP kode in upravlja s klientovimi zahtevki, ter mu vrača generiran HTML,
- podatkovni strežnik, ki skrbi za shranjevanje podatkov, zaklep zapisov, integriteto in varnost podatkov,
- odjemalca, ki je pri naši aplikaciji brskalnik in skrbi za prikaz uporabniškega vmesnika, ter ob uporabnikovi interakciji pošilja zahtevke aplikacijskemu strežniku.



Slika 4.1: Poenostavljen prikaz delovanja tri-nivojske arhitekture

4.2 Podatkovna baza

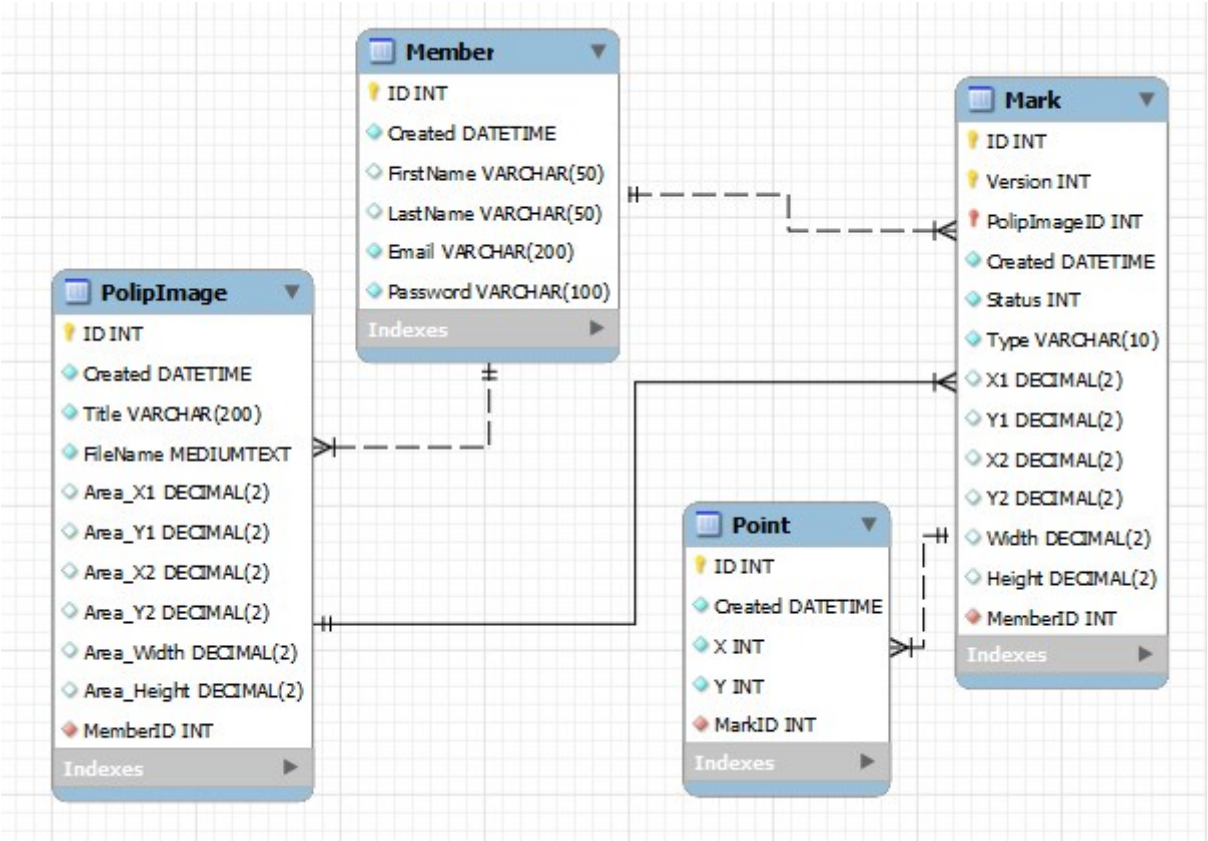
Eden izmed korakov izdelave spletne aplikacije je bila podatkovna baza. Potrebno je bilo analizirati vse funkcionalnosti in na njihovi osnovi zgraditi tabele. Samo bazo sem izdelal kar z namenskim orodjem phpMyAdmin, sestavljajo pa jo naslednje glavne entitete (slika 4.2):

- **Member (uporabnik)** – Tabela vsebuje osnovne informacije o uporabniku kot sta ime in priimek, ter unikaten e-naslov in geslo s katerima se uporabnik lahko prijavi v sistem. V tabelah PolipImage in Mark, preko relacije na polje MemberID, definira kdo je ustvaril sliko oz. oznako.

- **PolipImage (slika polipa)** – Objekti tabele se ustvarijo ob nalaganju novih slik preko sistema za masovno nalaganje. Zapiše se ime datoteke, na podlagi katerega se ustvari tudi privzeti naslov. Ker se za vsako sliko določa tudi območje, v katerem se polipi iščejo, smo ustvarili tudi polja s koordinatami x, y, ki povedo od kod, ter do kam sega območje, ter njegovo širino in višino. MemberID skrbi za povezavo s tabelo uporabnika, da lahko vemo kdo je sliko naložil.

- **Mark (oznaka)** – Entiteta oznaka skrbi za hranjenje podatkov o tem kje se nahaja polip. Pozicija je definirana z začetnima in končnima koordinatama x, y, ter višino in širino. Dodatno jo definira še tip (krog, elipsa, kvadrat, pravokotnik, poligon), status (potrjen, zavrnjen, nedefiniran), datum nastanka in kdo jo je ustvaril. Vsaka oznaka je tudi del verzije, pripada določeni sliki in ima unikaten ID. Te tri karakteristike skupaj tudi tvorijo ključ, preko katerega je možno spremljati, kaj se z neko oznako dogaja skozi verzije.

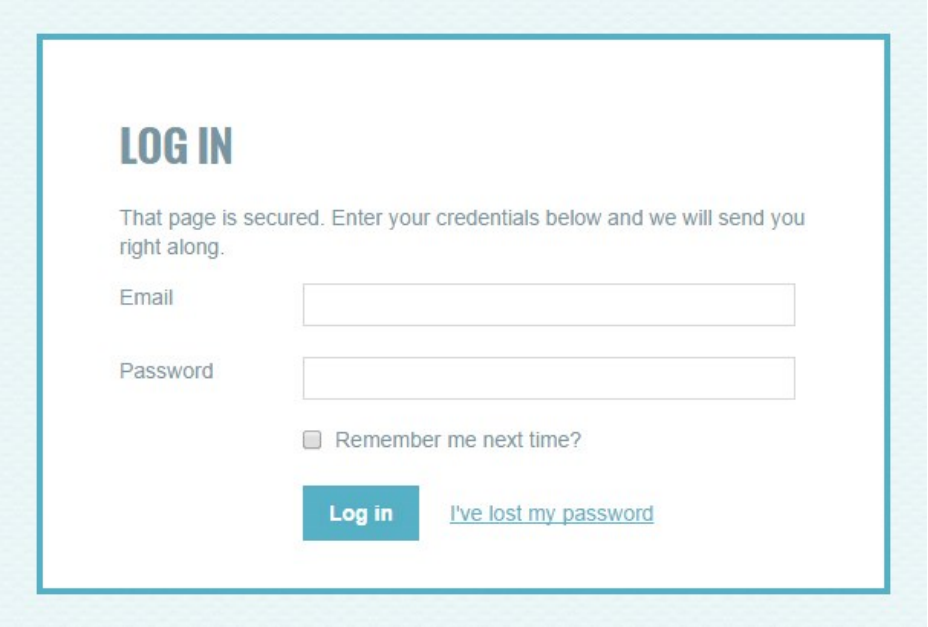
- **Point (točka)** – V primeru, ko je oznaka tipa poligon, je ne moremo definirati s pozicijo in dimenzijami, ampak z množico točk, ki ta poligon sestavljajo. Tako se v tej tabeli hranijo vse koordinate x, y posameznih vozlišč poligonov.



Slika 4.2: Podatkovni model aplikacije

4.3 Prijava v aplikacijo

Aplikacija kot taka ni namenjena javni uporabi, zato se mora uporabnik pred uporabo prijaviti v sistem s svojim uporabniškim imenom (e-poštni naslov) in geslom (slika 4.3), ki se preverita v podatkovni bazi. Uporabnik je v tem primeru lahko bodisi zaposleni na Morski biološki postaji Piran, bodisi zunanji strokovnjak na tem področju. Pred prijavo pa ga mora skrbnik dodati v sistem.



Slika 4.3: Prijavni obrazec v spletno aplikacijo

Če je uporabnik vnesel napačno kombinacijo e-poštnega naslova in gesla, mu aplikacija to ustrezno sporoči. Zgodi se lahko, da naše geslo pozabimo, zato imamo na voljo tudi povezavo, ki nam odpre obrazec, kamor vnesemo naš poštni naslov in če se ta ujema s tistim v podatkovni bazi, kmalu za tem prejmemo ponastavljeno geslo.

Da uporabniku ne bi bilo potrebno vsakič, ko želi uporabljati aplikacijo, vnašati uporabniškega imena in gesla, lahko označi naj si ga sistem zapomni. V tem primeru se ustvari naključni uporabniški žeton (daljše zaporedje znakov), ki se zapiše v bazo in v piškotek brskalnika. Pri naslednjem poskusu dostopa lahko uporabnika ob ujemanju vrednosti piškota in vrednosti v bazi avtomatično prijavimo.

Po uspešni prijavi, nas aplikacija privzeto preusmeri na osnovno stran, če pa smo poskušali dostopati do katere druge zaklenjene strani, si je sistem to zapomnil (shranil kot URL parameter »BackURL«) in nas bo preusmeril na željeno stran.

4.4 Seznam slik

Seznam slik (slika 4.4) je osrednja stran za upravljanje naše aplikacije. Primarne funkcionalnosti so pregled, dodajanje novih in odstranjevanje obstoječih slik. Omogoča pa tudi filtriranje po datumu in številčenje strani, določanje območja na fotografiji, pregled

verzij, proženje avtomatskega zaznavanja polipov, preimenovanje slik in prikaz dodatnih informacij iz fotoaparata.

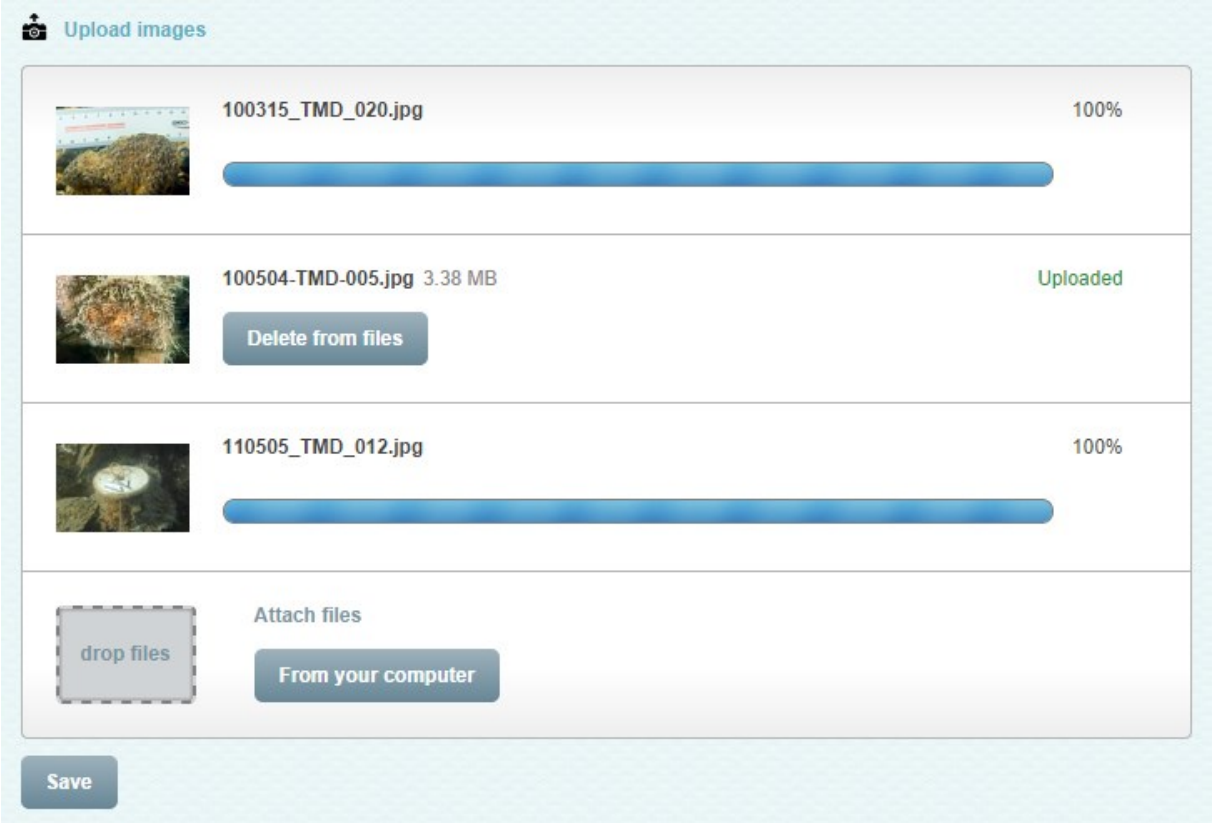
Slika 4.4: Seznam naloženih fotografij polipov

4.4.1 Masovno nalaganje slik

Ko potapljač poslika polipe, je fotografije za obdelavo potrebno najprej naložit v aplikacijo. To naredimo tako, da fotoaparat priklopimo na računalnik, prenesemo slike v datotečni sistem (slika 4.5), odpremo aplikacijo in se prijavimo, nato pa jih lahko kar po sistemu povleci in spusti potegnemo v brskalnik na za to predvideno polje. Če imamo starejši brskalnik (Firefox 3, Opera 11, Internet Explorer 9, Safari 4), ta opcija ni na voljo, še vedno pa lahko slike naložimo tako, da jih v oknu za izbiro slik poiščemo in izberemo. Za nalaganje slik skrbi knjižnica jQuery File Upload [10], ki ima za enostavno nalaganje kar nekaj zanimivih funkcionalnosti:

- možnost hkratnega nalaganja slik,
- sprotni prikaz napredka nalaganja posamezne datoteke,
- podprt sistem povleci in spusti,
- prekinitev prenosa posamičnih datotek,
- podporo za nadaljevanje prekinjenih prenosov od tam kjer so se končali,

- brskalnik ne potrebuje dodatnih vtičnikov kot je npr. Adobe Flash, saj je prenos podprt s HTML5,
- predogled slik ob nalaganju,
- v primeru starejšega brskalnika še vedno deluje, vendar brez zgoraj naštetih izboljšav.



Slika 4.5: Vmesnik za masovno nalaganje slik

Po uspešnem nalaganju slike še potrdimo s klikom na gumb shrani, kar ustvari primerne zapise v podatkovni bazi, pripravi sličice za predogled in določi privzeto območje za iskanje polipov na celotno sliko.

Privzeto se naslov fotografije ustvari iz samega imena datoteke. Na seznamu se zraven vsakega naslova prikaže ikona svinčnika, ob kliku nanjo pa naslov spremeni v polje za urejanje. Spremembo naslova je po tem potrebno samo še potrditi, s klikom na kljukico. Nov naslov se v ozadju kot AJAX pošlje na strežnik in zapiše v bazo.

4.4.2 Filtriranje in navigacija po slikah

Po daljšem času uporabe aplikacije se bo gotovo nabrala večja količina slik. Ker vseh ne moremo prikazati na eni strani, saj bi bilo to zelo neučinkovito in nepregledno, smo morali prikaz slik omejiti. Tako se na eni strani prikazuje le deset slik. Slike so razvrščene datumsko padajoče od najnovejše pa do najstarejše. Za dostop do ostalih strani smo pod seznamom implementirali številčenje strani, ki omogoča prehod na prejšnjo ali naslednjo stran, kot tudi preskok na poljubno drugo stran.

Številčenje strani za delovanje uporablja URL parameter »start«, ki ga aplikacija v programski kodi PHP prebere ter izračuna od katere slike naprej mora iz baze naložiti preostale slike. Npr. če ima start vrednost 30, pomeni da bomo naložili slike od tridesete naprej pa do štiridesete.

Včasih se zgodi, da iščemo neko starejšo fotografijo, za katero ne vemo na kateri strani se bo nahajala, poznamo pa približno obdobje v katerem je bila posneta. Ker jo lahko po takšnem ključu precej hitreje poiščemo, smo razvili obrazec za filtriranje slik po datumskem razponu (slika 4.6).

Slika 4.6: Filtriranje slik po datumu s pomočjo izbire datumov iz koledarčka

Kot je razvidno iz slike, imamo dve vnosni polji – »datum od« in »datum do«. Ob kliku v vnosno polje smo za lažjo uporabo, da ni potrebno vpisovati datuma, dodali izbirni koledarček, ki ga omogoča knjižnica jQuery UI Datepicker [11]. Po potrditvi se obrazec z

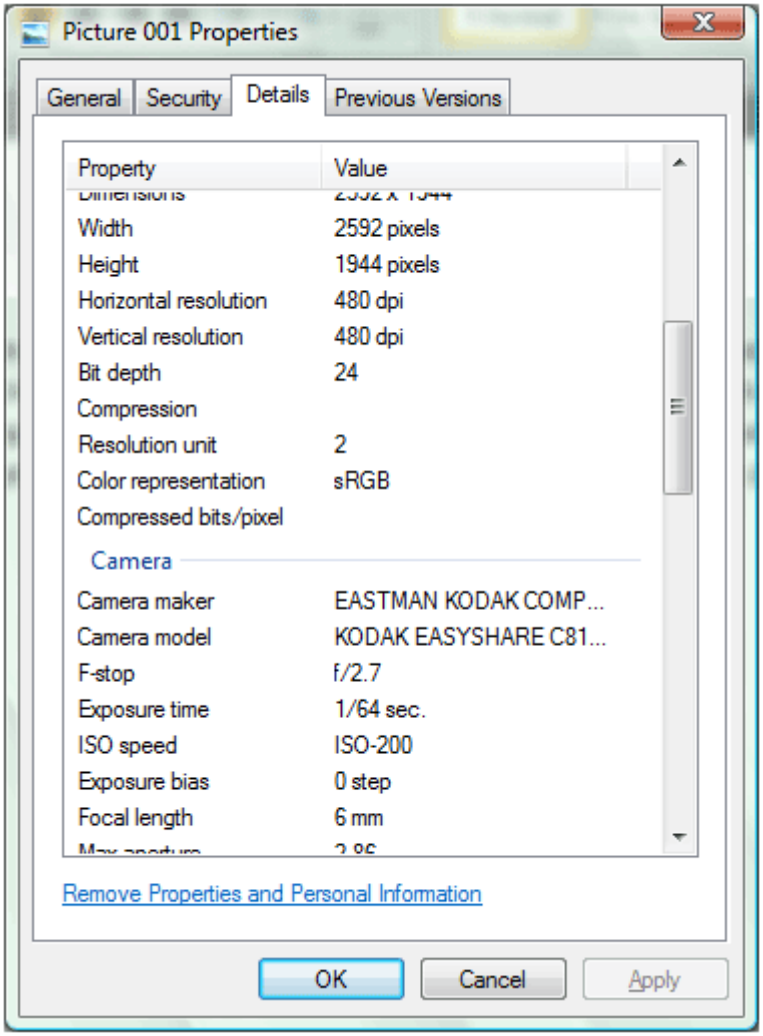
datumoma pošlje na strežnik, nabor slik pa se zmanjša na vse, ki so bile posnete v tem razponu. Številčenje strani sedaj deluje znotraj tega nabora slik.

4.4.3 EXIF podatki o sliki

Ob pregledu fotografij so si fotografi že od nekdaj želeli podoživeti kako je bil nastavljen fotoaparat ob zajetju motiva, kakšna je bila svetloba in kakšni so bili pogoji na samem kraju. S pojavom digitalnih kamer se je tako pojavil EXIF – skupek prenosljivih podatkov o fotografiji, ki je specifikacija za slikovni datotečni format po kateri se k sliki zapišejo še različni metapodatki (slika 4.7), med katerimi so:

- proizvajalec in model fotoaparata,
- datum in ura nastanka fotografije,
- dimenzije in resolucija slike,
- ali je bila uporabljena bliskavica,
- GPS koordinate lokacije slikanja,
- informacije o zaslonki in svetlobi …

EXIF podatki so v našem primeru potrebni pri nadaljnji (avtomatski) analizi slik in štetju polipov.



Slika 4.7: Pregled EXIF podatkov o sliki v Windows File Explorerju

Za zajem EXIF podatkov iz slike ima PHP že pripravljeno funkcijo exif_read_data [5]. V seznamu sicer prikazujemo le osnovne podatke, kot so kdaj je bila slika posneta, s katerim fotoaparatom in katerega tipa je slika. Če je bila zajeta tudi lokacija, zemljepisno širino in dolžino pretvorimo v decimalni format in uporabniku ponudimo povezavo, s katero si lokacijo lahko ogleda na Googlovih zemljevidih.

4.4.4 Označevanje polipov

Ko se fotografija naloži na strežnik, je pripravljena za iskanje polipov na njej. Ker je aplikacija zamišljena kot pripomoček za popravljanje avtomatsko zaznanih oznak, je potrebno vse spremembe zabeležiti v podatkovno bazo, iz katere bomo lahko zagotovili strojno učenje avtomatskega zaznavanja. Zanima nas celotna zgodovina oznak na fotografiji, zato moramo

vsako spremembo oznake shraniti kot novo verzijo. Pri ustvarjanju verzij veljajo naslednja pravila:

- iskanje in urejanje oznak najdenih polipov vedno poteka po izbranem območju,
- prvo verzijo ustvarimo s poganjanjem avtomatskega iskanja in je zaklenjena za urejanje, saj bi sicer izgubili podatke o avtomatsko zaznanih oznakah,
- za avtomatsko verzijo je mogoče ustvariti uporabniško, ki je dejansko kopija avtomatske in jo je mogoče urejati,
- brišemo lahko le uporabniške verzije ne pa tudi avtomatskih, vendar le tiste, za katere ne obstaja novejša avtomatska verzija.

V času pisanja drugi del aplikacije, to je avtomatsko zaznavanje polipov in ustvarjanje oznak še ni razvit, zato je narejena funkcija (slika 4.8), ki simulira avtomatiko tako, da naključno ustvari eno do deset oznak, pri čemer oznaka ni nikoli poligon, za čisto ročno delo pa bi lahko ustvarili prazno množico oznak.

```php
function generateRandomMarks() {
    // objekt slike nad katero bomo ustvarili naključne oznake
    $polipImage = $this->currentPolipImage();
    if (!$polipImage) return false;

    // preverimo če je zadnja verzija uporabniška
    $lastVersion = $polipImage->lastVersion();
    if ($lastVersion && $lastVersion->Type == 'system') return false;

    // zaporedna številka naslednje verzije
    $nextVersion = $polipImage->nextVersion();

    // nabor oblik oznak
    $types = array('Circle', 'Ellipse', 'Square', 'Rectangle');

    // ustvarimo 10 naključnih oznak
    for ($i = 1; $i <= rand(1, 10); $i++) {
        $mark = new Mark();

        // osnovne lastnosti
        $mark->ID = $i;
        $mark->Version = $nextVersion;
        $mark->PolipImageID = $polipImage->ID;

        // naključni status in tip oznake
        $mark->Type = $types[array_rand($types, 1)];
        $mark->Status = rand(-1, 1);

        // naključna širina in višina (za kvadrat in krog enaki)
        $mark->width = rand(50, 200);

        if ($mark->Type == 'Circle' || $mark->Type == 'Square') {
            $mark->height = $mark->width;
        }
        else $mark->height = rand(50, 200);

        // naključna pozicija + varovalo da je znotraj območja
        $mark->x1 = rand(0, $polipImage->Area_width - $mark->width);
        $mark->y1 = rand(0, $polipImage->Area_height - $mark->height);

        $mark->x2 = $mark->x1 + $mark->width;
        $mark->y2 = $mark->y1 + $mark->height;

        // zapišimo oznako
        $mark->write();
    }
}
```

Slika 4.8: Naključno ustvarjanje desetih oznak v programskem jeziku PHP

Na seznamu imamo na pregleden način predstavljene zadnje verzije po fotografijah. Za vsako verzijo je prikazano kdo jo je ustvaril in koliko oznak vsebuje, zraven pa so izpisane povezave za pregledovanje in urejanje, ter gumbi za brisanje in ustvarjanje novih verzij.

4.5 Izbira območja slike

Ustvarjanje podvodnih fotografij je težavno opravilo, saj je vidljivost pod vodo slabša kot na kopnem, potapljača pa pri tem lahko ovira še motna voda, maska in razni morski tokovi, zato je na fotografiji pogosto zajeta še okolica in ne samo polipi. Ker nas območje okoli polipov ne zanima in zna biti za avtomatsko zaznavanje celo moteče, nam aplikacija omogoča izrez oziroma izbiro območja (slika 4.9) kjer se iskani polipi dejansko nahajajo.

Slika 4.9: Uporabniški vmesnik za izbiro območja iskanja polipov na fotografiji

Vmesnik za izbiro območja temelji na knjižnici imgAreaSelect [8], ki za delovanje uporablja še JavasScript knjižnico jQuery. Kot vhod potrebuje le željeno sliko, ki je za potrebe zaslonskega prikaza pomanjšana na velikost, primerno za brskalnik. Omogoča nam:

- spreminjanje izgleda s pomočjo CSS,
- možnost upravljanja s tipkovnico (premikanje in raztezanje izbranega območja),
- funkcije, ki se prožijo ob različnih dogodkih (inicializaciji, začetku in koncu označevanja, spremembi označenega področja),
- API (Application Programming Interface) za enostavno integracijo z ostalimi komponentami aplikacije,

- veliko nastavitev obnašanja, kot so fiksno razmerje dimenzij, ali se naj ročke za raztezanje prikažejo, ali je omogočeno premikanje, izbrano območje ob inicializaciji, nastavljanje maksimalnih dimenzij …

```
var area = areaImgOriginal.imgAreaSelect({
    // prikaži ročke za raztezanje s pomočjo miške
    handles: true,

    // funkcija ki izvrši vse potrebno ob spremembi področja
    onSelectChange: areaOnSelectChange,

    // funkcija, ki ob inicializaciji naloži obstoječe stanje
    onInit: areaInit,

    // vrni instanco, da bomo lahko uporabljali API
    instance: true
});
```

Slika 4.10: Inicializacija JavaScript knjižnice za izbiro območja slike

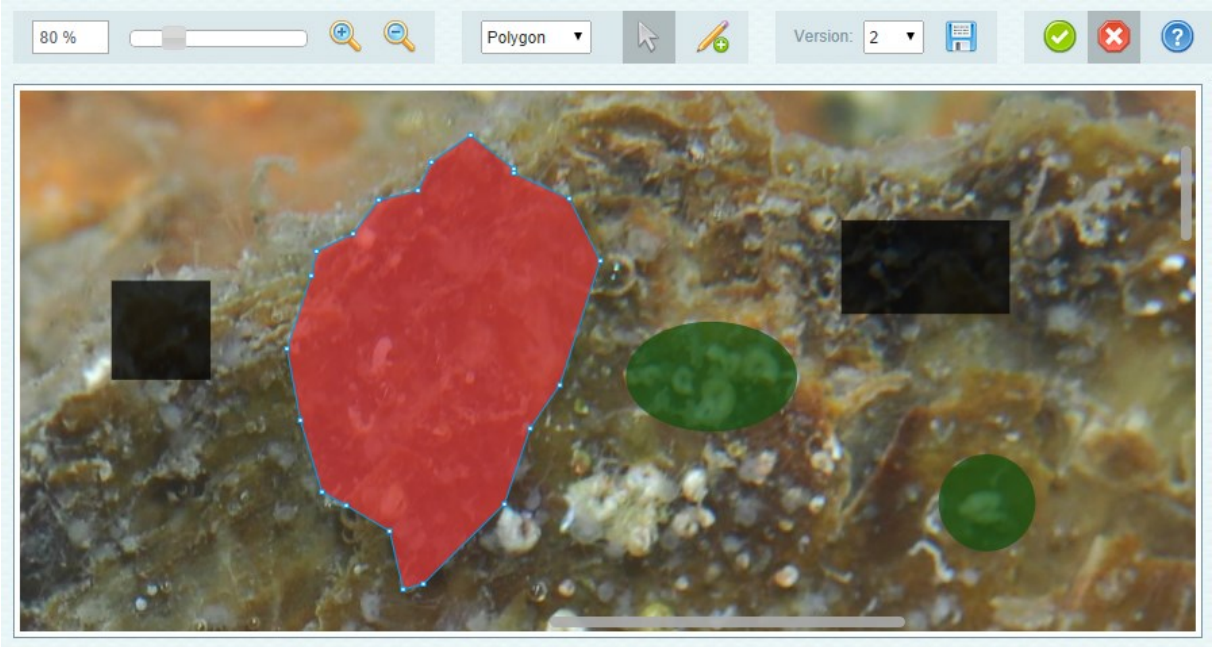
Po uspešnem nalaganju slike se kot začetno delovno območje vedno določi celotno območje fotografije. Ko se naloži vmesnik za spreminjanje območja, se iz podatkovne baze vedno naloži zadnje definirano stanje. Območje je mogoče spreminjati le, dokler se ne ustvari prva verzija oznak nad polipi, potem se dostop do tega dela aplikacije za to sliko onemogoči. Ob strani se pri vsaki spremembi spremenita tudi izpisani koordinati x, y od začetne do končne točke območja, izračuna pa se tudi širina in višina. Vsi ti podatki se preračunajo glede na razmerje med pomanjšano in izvirno velikostjo slike.

Ob kliku na gumb shrani se zajamejo vsi podatki, ki so izpisani na desni strani, ter se s pomočjo jQuery knjižnice v ozadju pošljejo kot AJAX (asinhroni JavaScript in XML) na strežnik, kjer se zapišejo v podatkovno bazo. Gumb nazaj nas pelje na seznam slik.

4.6 Urejevalnik oznak

Urejevalnik oznak (slika 4.11) je orodje za delo z oznakami na podvodnih fotografijah. Sestavljen je iz dveh delov, in sicer orodne vrstice in izbranega območja fotografije. Orodna vrstica je razdeljena na štiri tematske sklope:

- spreminjanje nivoja povečave fotografije,
- orodja za izbiro, kreiranje, ter izbor in spreminjanje tipa oznak,
- prikaz, spreminjanje in shranjevanje verzij,
- prikaz trenutnega stanja izbrane oznake, ter potrjevanje in zavračanje oznak.



Slika 4.11: Urejevalnik oznak polipov na podvodnih fotografijah

4.6.1 Orodje za povečavo

Privzeto se slika izbranega območja prikaže v naravni velikosti, in sicer v platnu z dimenzijami 950px širine in 400px višine. Dandanes modernejši fotoaparati zajemajo slike precej višjih ločljivosti, zato je realno pričakovati, da se ne prikazujejo primerno v tako majhnem polju, katerega dimenzije so prilagojene tako, da ustrezajo večini današnjih zaslonov. Ker so polipi majhni in jih je na zajeti fotografiji lahko veliko, bi bilo označevanje zelo težko in nenatančno, zato smo pri izdelavi aplikacije poskrbeli za možnost povečave. V primeru, ko slika presega okvire polja, je posledično potrebno poskrbeti tudi za premikanje prikazanega dela slike, kar je zagotovljeno s spodnjim in stranskim drsnikom.

V vsakem trenutku je v orodni vrstici prikazan trenutni nivo povečave, izražen v odstotkih (100% pomeni brez povečave). Spreminjamo ga lahko na tri načine – z drsnikom, s klikom na povečevalno lupo, ali pa uporabimo kombinacijo stiska tipke CTRL na tipkovnici in vrtenja koleščka na miški.

Za delovanje skrbi kombinacija treh JavaScript knjižnic. Za drsnik v orodni vrstici skrbi knjižnica jQuery UI Slider, za drsnika delovnega polja knjižnica Perfect scrollbar, možnost spreminjanja z miškinim koleščkom pa nam omogoča jQuery Mouse Wheel. Za vsak sprožen dogodek katerekoli knjižnice je potrebno izračunat nov nivo povečave, posodobiti vrednost v orodni vrstici in premakniti center pogleda delovnega polja s knjižnico Paper.js.

4.6.2 Verzije oznak polipov

V osnovi poznamo dva tipa verzije – avtomatsko in uporabniško. Pri avtomatski verziji je ročno urejanje vedno onemogočeno, kar pomeni, da oznak ne moremo premikati, dodajati, jim spreminjati velikosti, brisati in nasploh kakorkoli modificirati. V tem primeru urejevalnik deluje v načinu pogleda (angl. view mode). Če želimo na avtomatski verziji delati popravke, je potrebno najprej iz nje ustvariti uporabniško, ki je kopija avtomatske in vsebuje kopije oznak, ki imajo enake lastnosti, kar pomeni enake pozicije, dimenzije, ID, tip oblike in stanje. Na ta način lahko vedno vidimo, kakšne spremembe so bile ročno narejene. Tako se npr. nekemu elementu lahko spremeni tip oblike, drugega ni več (je izbrisan), tretji je bil povečan, četrti ima spremenjeno stanje, dodan pa je tudi kakšen nov. Takšna zgodovina služi kot baza znanja in je podlaga za strojno učenje.


```
[
   {
      "Version":"1",
      "MarkID":"1",
      "Status":"0",
      "Type":"Square",
      "x1":"1402.00",
      "y1":"449.00",
      "x2":"1457.00",
      "y2":"504.00",
      "width":"55.00",
      "height":"55.00",
      "PolipImageID":"15"
   }
]
```


Slika 4.12: Primer oznake definirane v JSON formatu

Nalaganje verzije poteka tako da, ko se urejevalnik inicializira, dobi v kodi tudi spremenljivko loadJSON, ki je, kot ime nakazuje, tipa JSON (JavaScript Object Notation) (slika 4.12). V njej se nahajajo vsi že zgoraj našteti podatki vseh oznak, ki jih potem preko Paper.js knjižnice ustrezno ustvarimo in prikažemo v delovnem polju. Proces shranjevanja je zelo podoben, lahko bi rekli tudi obraten, saj namesto kreiranja oznak preberemo njihove lastnosti in jih pretvorimo v JSON, ki se potem pošlje na strežnik. Sprožimo ga s klikom na ikono diskete. Orodje za delo z verzijami ima tudi polje s spustnim seznamom, s katerim preklapljamo med vsemi možnimi verzijami, ki so na voljo za trenutno fotografijo.

4.6.3 Urejanje oznak

Za delo z oznakami imamo na voljo dva osnovna orodja - orodje za izbiro (ikona miškinega kazalca) in orodje za dodajanje novih oznak (ikona svinčnika). Glede na izbrano orodje lahko na sliki počnemo različne stvari. Zraven se nahaja še spustni seznam z vsemi možnimi oblikami oznak – krog, elipsa, kvadrat, pravokotnik in poligon.

Novo oznako torej ustvarimo z izbiro primerne oblike in svinčnika, kliknemo na izhodišče in povlečemo. Če ustvarjamo krog ali kvadrat, sta širina in višina zaklenjeni na razmerje ena proti ena. V primeru, ko ustvarjamo poligon, miške ne vlečemo, ampak s kliki na vogale ustvarjamo vozlišča.

V primeru, ko delamo z orodjem za izbiro, nam ta omogoča:

- izbiro oznake – oznaka je izbrana, ko se okoli nje pojavi moder okvir in se prikažejo vse ročke (angl. handles),
- premikanje oznake po fotografiji – oznako premikamo tako, da jo označimo, potem pa po sistemu »povleci in spusti« z miško določimo njeno novo pozicijo,
- brisanje – samo če je oznaka izbrana, jo lahko izbrišemo s pritiskom tipke »briši« na tipkovnici,
- spreminjanje stanja oznake – oznake imajo tri možna stanja:
    - potrjena – uporabnik se strinja, da se na označenem mestu resnično nahaja polip, zato v orodni vrstici izbere ikono z zeleno kljukico in oznako potrdi, ta pa prav tako postane zelena,
    - zavrnjena – uporabnik je ugotovil, da se na označenem mestu vsekakor ne nahaja polip, zato izbere rdečo ikono s križcem, oznaka postane rdeča,

  - nedefinirana (privzeto stanje oznake) - uporabnik ni prepričan, ali je na označenem mestu res polip ali ne,

- spreminjanje tipa oblike oznake – tip trenutno izbrane oznake spremenimo tako, da iz spustnega seznama izberemo drugega, kjer veljajo naslednja pravila:

  - če obliko spreminjamo iz pravokotnika, elipse ali poligona v kvadrat ali krog, bo dolžina stranice enaka večji stranici mejnega pravokotnika, ki zajema element,

  - če katerokoli obliko spreminjamo v poligon, bo ta dobil obliko pravokotnika,

  - če spreminjamo poligon v pravokotnik ali elipso, bodo dimenzije enake dimenzijam mejnega pravokotnika poligona,

- spremembo velikosti oznake – s klikom na ročko in njenim premikanjem je možno oznako povečati ali pomanjšati, kjer pa veljajo iste zakonitosti kot pri risanju nove.

Glede na aktivnost miške oziroma kurzorja znotraj delovnega polja, se z uporabo knjižnice Paper.js v osnovi prožijo trije dogodki (funkcije), ki zagotavljajo zgoraj opisane funkcionalnosti:

- stisk levega gumba (onMouseDown)

  - pri ustvarjanju zabeleži koordinati x, y, ki ob kasnejšem premikanju služita kot začetna točka novo ustvarjene oznake, v primeru ko ustvarjamo poligon, pa doda novo vozlišče,

  - pri izbiranju preverja na kaj smo kliknili in v primeru, ko gre za oznako, le-to izbere (naredi označeno), ko pa smo kliknili na ročko, si zapomni za katero gre, da lahko ob premiku ustrezno spremenimo velikost,

- vleka miške ob pritisnjenem levem gumbu (onMouseDrag)

  - pri ustvarjanju oznake ob vsakem premiku staro oznako izbriše in nariše novo od začetne točke pa do točke, kjer se trenutno nahaja kurzor (slika 4.13),

  - pri izbiranju pozicijo elementa prestavi na trenutno pozicijo kurzorja, če pa premikamo ročko, ustrezno spremeni velikost oznake, oziroma v primeru poligona samo prestavi pozicijo vozlišča,

- spust levega gumba (onMouseUp) – skrbi da oznaka postane neoznačena.

```
newTool.onMouseDrag = function(event) {
    // poligon ustvarjamo s klikanjem, ne vlečenjem
    if (!path || pathIsPolygon()) return;

    // shrani lastnosti oznake in jo izbriši
    var data = path.data;
    pathRemove();

    // izračunaj novo pozicijo in dimenzije oznake
    var x = Math.min(event.point.x, cursorX),
        y = Math.min(event.point.y, cursorY),
        w = Math.abs(event.point.x - cursorX),
        h = Math.abs(event.point.y - cursorY);

    // v primeru kvadrata ali kroga naj bosta dimenziji enaki večji
    if (data.type == 'Circle' || data.type == 'Square') {
        if (w > h) h = w;
        else w = h;
    }

    // poustvari oznako s starimi lastnostmi in novimi dimenzijami
    pathCreate(new Rectangle(x, y, w, h), data);
}
```

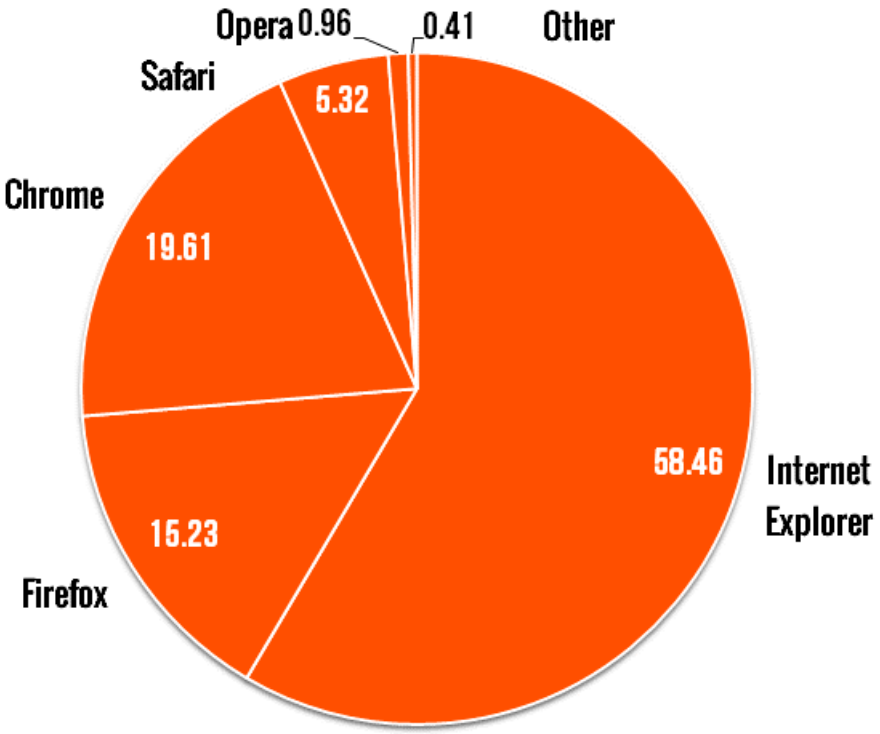
Slika 4.13: Funkcija za ustvarjanje oznake, ki se proži ob vleki miškinega kazalca

Poglavje 5 Testiranje aplikacije

Pred začetkom resne uporabe je potrebno vsako aplikacijo čim bolj temeljito testirati, da kasneje ne pride do večjih težav. Testiranje funkcionalnosti je sicer potekalo že med samim razvojem aplikacije, zato ni bilo pričakovati večjih problemov. Pred končnim testiranjem bi bilo dobro aplikacijo iz lokalnega razvojnega okolja namestiti še v delovno okolje. Ker dejansko okolje še ni bilo na voljo, sem uporabil kar enega izmed zastonjskih gostovanj, ki so na spletu.

5.1 Različni brskalniki

Dandanes je v uporabi mnogo različnih spletnih brskalnikov, ki si jih uporabniki lahko namestijo, zato je bil potreben test delovanja vsaj v najpopularnejših (slika 5.1), kot so Chrome, Firefox, Internet Explorer, Safari in Opera.



Slika 5.1: Odstotek uporabe različnih namiznih brskalnikov za avgust 2014 [21]

Pri tem tipu testiranja se postavlja v ospredje predvsem test izgleda uporabniškega vmesnika, ki je na žalost po navadi v vsakem brskalniku nekoliko drugačen. Do tega prihaja, ker nekateri brskalniki (gre predvsem za starejše različice) ne podpirajo oz. različno interpretirajo nekatera določila zadnjih različic CSS. Vse odkrite napake smo tako odpravili, ni pa bilo mogoče v celoti zagotoviti popolnoma identičnega izgleda, saj nekatere elemente brskalniki prikazujejo tako kot jih in ne obstajajo nastavitve, ki bi to spremenile. V večini gre predvsem za pisave in elemente obrazcev, kot je npr. spustni meni.

5.2 Delovanje aplikacije na strežniku

Razvoj je potekal lokalno na operacijskem sistemu Windows ob podpori strežnika IIS (Internet Information Services). Ciljna platforma za aplikacijo je Linux in strežnik Apache, zato smo za test aplikacije poiskali prav takšno zastonjsko gostovanje, ki to omogoča. Na srečo tukaj ni prišlo do nobenega problema. Edini problem, ki smo ga zaznali, je pojavljanje napak pri nalaganju fotografij večjih dimenzij, saj se v tem trenutku ustvari tudi pomanjšana sličica za predogled, zato je treba sliko pomanjšat. Za to skrbi PHP knjižnica GD, ki je očitno pri večjih dimenzijah slik precej spominsko požrešna, zato je v nekaj primerih prišlo do pomanjkanja pomnilnika. Ker gre za zastonjsko gostovanje, tega problema nismo mogli rešiti, saj ne omogoča spreminjanja omejitve dodeljenega spomina, ki bi ga bilo potrebno povečati v PHP konfiguracijski datoteki.

5.3 Varnost aplikacije

Ena izmed zahtev pri izdelavi aplikacije je bila, da ni javno dostopna vsem, ampak samo avtoriziranim uporabnikom, zato smo še enkrat testirali prijavo v aplikacijo z različnimi uporabniškimi računi, takšnimi ki obstajajo in takšnimi, ki ne. Prav tako smo poskušali dostopati do različnih delov aplikacije neprijavljeni. Preverili smo, da je dostop do podatkovne baze omogočen samo strežniku in ustrezno zaščiten z uporabniškim imenom in geslom. Ker naša aplikacija vsebuje tudi nekatera vnosna polja, iz katerih zajema podatke, s katerimi potem operira nad bazo, smo bili pozorni, da ni mogoče vriniti kakšnega SQL stavka, kar je zelo popularen napad, imenovan SQL injection [18]. Za zagotavljanje varnosti tudi v bodoče, bodo pomembne sprotne in redne nadgradnje vseh vpletenih tehnologij.

5.4 Uporabniška izkušnja

Pri izdelavi aplikacije smo se trudili, da bi bil uporabniški vmesnik čimbolj preprost, intuitiven in enostaven za uporabo. Že pri samem izgledu smo se glede na to, da delamo s

podmorskimi fotografijami, uporabniku želeli približati tako, da smo za dizajn uporabili modre barve ozadij in teksta. Prednosti uporabniškega vmesnika so še:

- prijavni obrazec vsebuje enaka polja in izgled večine prijavnih obrazcev na spletu,
- za enostavnost nalaganja slik smo implementirali sistem povleci in spusti, ki so ga uporabniki že navajeni pri prenašanju datotek iz različnih zunanjih naprav in medijev,
- pri seznamu smo uporabniku ponudili predogled s sličicami in osnovne informacije o sliki in verzijah,
- da ga ne bi obremenili s preveliko količino slik, smo jih razdelili na več strani in omogočili številčenje strani,
- urejevalnik smo zasnovali tako, da je čim bolj podoben namiznim aplikacijam za urejanje slik – z vrstico z že znanimi ikonami posameznih orodij in delovnim poljem, ki omogoča tudi povečavo.

Sam test vmesnika je bil izveden kar v družinskem krogu. Izkazalo se je, da uporabniki po razlagi samega namena aplikacije, niso imeli večjih težav z uporabo. Dejstvo pa je, da bi vseeno bilo potrebno priložiti navodila, kjer bi bili razloženi postopki (ustvarjanje verzij) in skrite funkcije aplikacije, kot sta npr. brisanje s tipko delete in povečevanje z miškinim koleščkom ob pritisku na tipko CTRL.

Poglavje 6 Zaključek

Cilj diplomskega dela je bil razvoj aplikacije, namenjene morskim biologom, ki se ukvarjajo z raziskavo števila polipov v morju. Med zadanimi cilji so bili nalaganje in urejanje fotografij in oznak polipov, na uporabniku prijazen način.

Razvoj aplikacije je potekal načrtovano in premišljeno. Ob izdelavi diplomskega dela sem utrdil znanja že poznanih spletnih tehnologij in spoznal nove, kot sta HTML element canvas in knjižnica Paper.js, ki sta mi bili v veliki meri neznanka.

Menim, da nam je v nalogi uspelo uresničiti vse zastavljene cilje. Uspešno smo implementirali obrazce, omogočili enostavno hkratno nalaganje več slik in njihov pregled, zajeli smo dodatne informacije, ki jih slikam pripiše fotoaparat, največji izziv pa je predstavljal razvoj urejevalnika oznak polipov in beleženje zgodovine sprememb skozi verzije, ki bodo v prihodnosti omogočile strojno učenje in avtomatsko detekcijo.

Končni izdelek diplomskega dela je tako sicer delujoča spletna aplikacija, ki pa za uporabo v delovnem okolju potrebuje še njen drugi del, to je avtomatsko iskanje polipov.

6.1 Možnosti za nadaljnji razvoj

V spletni aplikaciji smo sicer implementirali vse zadane cilje, vendar je možnosti za nadaljnji razvoj verjetno še kar nekaj. Dejanske potrebe pa se bodo pokazale šele ob uporabi v delovnem okolju na Morski biološki postaji Piran.

Da bi aplikacija lahko zaživela in obrodila sadove, je potrebna še implementacija avtomatskega zaznavanja polipov, kar se bo realiziralo kot druga diplomska naloga. Za uspešno povezovanje obeh delov je tako potrebno:

- spremeniti klic funkcije avtomatskega zaznavanja, ki trenutno ustvari naključne oznake polipov, na dejansko aplikacijo, ki bo to počela in je lahko narejena v kakšnem drugem programskem jeziku,
- aplikaciji je potrebno kot vhodni parameter poslati ID fotografije, nad katero izvajamo avtomatsko iskanje,

- aplikacija se mora znati povezati na podatkovno bazo MySQL in prebrati obstoječe oznake in njihove verzije,
- iz baze mora prebrati pot do fotografije in območje iskanja, ter dejansko zagnati algoritem iskanja polipov ter oznake zapisati nazaj v bazo kot novo verzijo,
- naši spletni aplikaciji naj sporoči, da je s procesiranjem končala, mi pa bomo osvežili seznam in prikazali novo verzijo in podatke o oznakah,
- ker ima dostop do baze, ima kadarkoli na voljo vse podatke, ki jih potrebuje za strojno učenje in izpopolnjevanje zaznavanja.

Zgoraj opisani postopek bi bilo najbolje implementirati v obliki REST (Representational state transfer) storitve.

Literatura

[1] (2014) Canvas element. Dosegljivo: http://en.wikipedia.org/wiki/Canvas_element

[2] (2014) CSS. Dosegljivo: http://sl.wikipedia.org/wiki/CSS

[3] (2014) Eclipse IDE. Dosegljivo: http://en.wikipedia.org/wiki/Eclipse_(software)

[4] (2014) Exchangeable image file format. Dosegljivo: http://en.wikipedia.org/wiki/Exchangeable_image_file_format

[5] (2014) exif_read_data(). Dosegljivo: http://php.net/manual/en/function.exif-read-data.php

[6] (2014) Firebug. Dosegljivo: http://getfirebug.com/whatisfirebug

[7] (2014) HTML. Dosegljivo: http://sl.wikipedia.org/wiki/HTML

[8] (2014) imgAreaSelect. Dosegljivo: http://odyniec.net/projects/imgareaselect/

[9] (2014) Internet Information Services. Dosegljivo: http://en.wikipedia.org/wiki/Internet_Information_Services

[10] (2014) jQuery File Upload. Dosegljivo: https://blueimp.github.io/jQuery-File-Upload/

[11] (2014) jQuery UI Datepicker. Dosegljivo: http://jqueryui.com/datepicker/

[12] (2014) jQuery. Dosegljivo: http://jquery.com/

[13] (2014) LAMP. Dosegljivo: http://en.wikipedia.org/wiki/LAMP_(software_bundle)

[14] (2014) Morska biološka postaja Piran. Dosegljivo: http://www.mbss.org/portal/index.php?option=com_content&task=view&id=1&Itemid=45

[15] (2014) MySQL. Dosegljivo: http://en.wikipedia.org/wiki/MySQL

[16] (2014) Paper.js. Dosegljivo: http://paperjs.org/about/

[17] (2014) PHP: Hypertext Preprocessor. Dosegljivo: http://en.wikipedia.org/wiki/PHP

[18] (2014) SQL Injection. Dosegljivo: http://php.net/manual/en/security.database.sql-injection.php

[19] (2014) Tri-nivojska arhitektura. Dosegljivo: http://colos1.fri.uni-lj.si/ERI/RACUNALNISTVO/PODATKOVNE_BAZE/trinivojska_arhitektura.html

[20] (2014) W3C Consortium. Dosegljivo: http://www.w3.org/Consortium/

[21] (2014) Worldwide desktop browser market share 08/2014. Dosegljivo: http://arstechnica.com/information-technology/2014/09/august-growth-puts-windows-8-back-on-track/